SCRUM DEVELOPER

CSD (5 Days)

Accentient's Scrum Developer course is a unique and intensive five-day experience for software developers. The course guides teams on how to turn product requirements into potentially shippable increments of software using the Scrum framework, Visual Studio 2010, and modern software engineering practices. Attendees will work in self-organizing, self-managing teams using a common instance of Visual Studio Team Foundation Server 2010 to achieve this goal.

AT COURSE COMPLETION

Scrum will be experienced through a combination of lecture, demonstration, discussion, and hands-on exercises. Attendees will learn how to do Scrum correctly while being coached and critiqued by the instructor, in the following topic areas:

- Form effective teams
- Explore and understand legacy "Brownfield" architecture
- Define quality attributes, acceptance criteria, and "done"
- Create automated builds
- How to handle software hotfixes
- Verify that bugs are identified and eliminated
- Plan releases and sprints
- Estimate product backlog items
- Create and manage a sprint backlog
- Hold an effective sprint review
- Improve your process by using retrospectives
- Use emergent architecture to avoid technical debt
- Use Test Driven Development as a design tool
- Setup and leverage continuous integration
- Use Test Impact Analysis to decrease testing times
- Manage SQL Server development in an Agile way
- Use .NET and T-SQL refactoring effectively
- Build, deploy, and test SQL Server databases
- Create and manage test plans and cases
- Create, run, record, and play back manual tests
- Setup a branching strategy and branch code
- Write more maintainable code
- Identify and eliminate people and process dysfunctions
- Inspect and improve your team's software development process

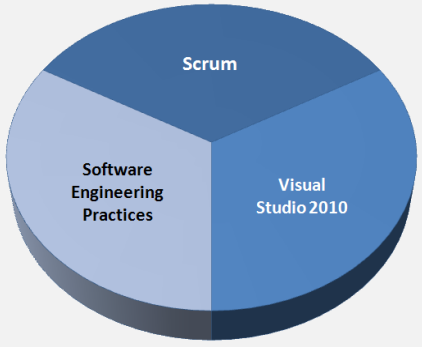
AUDIENCE

This course is suitable for any member of a software development team – architect, programmer, database developer, tester, etc. Entire teams are encouraged to attend and experience the course together, but individuals are welcome too.

Attendees will self-organize to form cross-functional Scrum teams. These teams require an aggregate of skills specific to the selected case study. Please see the last page of this document for specific details.

Product Owners, ScrumMasters, and other stakeholders are welcome too, but keep in mind that everyone who attends will be expected to commit to work and pull their weight on a Scrum team.

SUBJECT MATTER BREAKDOWN



THE WEEK AT A GLANCE

This course is a mix of lecture, demonstration, group discussion, simulation, and hands-on software development. The bulk of the course will be spent working as a team on a case study application delivering increments of new functionality in mini-sprints.

At a high level, here's the layout of the week:

	Monday	Tuesday	Wednesday	Thursday	Friday
Morning	Introduction Scrum Fundamentals Simulation	Brownfield Development Hotfix	Coding Topics Sprint 2	Testing Topics Sprint 4	Dysfunction
Afternoon	Implementing Scrum Case Study	Architecture Topics Sprint 1	Database Topics Sprint 3	(Optional) Sprint 5	Scrum FAQ

Monday morning and most of the day Friday will be spent with the computers powered off, so you can focus on sharpening your game of Scrum and avoiding common pitfalls when implementing it.

THE SPRINTS

Timeboxing is a critical concept in Scrum as well as in this course. We expect each team and student to understand and obey the timebox. The timebox duration will always be clearly displayed during an activity. Expect the instructor to enforce it.

Each of the ½ day sprints will roughly follow this schedule:

Component	Description	Minutes
Instruction	Presentation and demonstration of new and relevant tools & practices	60
Sprint planning meeting	Product owner presents backlog; each team commits to delivering functionality	10
Sprint planning meeting	Each team determines how to build the functionality	10
The Sprint	The team self-organizes and self-manages to complete their tasks	120
Sprint Review meeting	Each team will present their increment of functionality to the other teams	≤ 30
Sprint Retrospective	A group retrospective meeting will be held to inspect and adapt	10

Each team is expected to self-organize and manage their own work during the sprint. Pairing is highly encouraged. The instructor/product owner will be available if there are questions or impediments, but will be hands-off by default. You should be prepared to communicate and work with your team members in order to achieve your sprint goal. If you have development-related questions or get stuck, your partner or team should be your first level of support.

MODULE 1: INTRODUCTION

This module provides a chance for the attendees to get to know the instructors as well as each other. The Scrum Developer program, as well as the day by day agenda, will be explained. Finally, the Scrum team will be selected and assembled so that the forming, storming, norming, and performing can begin.

- Trainer and student introductions
- The Scrum Developer program
- Agenda
- Logistics
- Team formation
- Retrospective

MODULE 2: SCRUMDAMENTALS

This module provides a level-setting understanding of the Scrum framework including the roles, timeboxes, and artifacts. The team will then experience Scrum firsthand by simulating a multi-day sprint of product development, including planning, review, and retrospective.

- Scrum overview
- Scrum roles
- Scrum timeboxes (ceremonies)
- Scrum artifacts
- Simulation
- Retrospective

It's required that you read Ken Schwaber's Scrum Guide in preparation for this module and course.

MODULE 3: IMPLEMENTING SCRUM

This module demonstrates how to implement Scrum in Visual Studio 2010 using a Scrum process template*. The team will learn the mapping between the Scrum concepts and how they are implemented in the tool. After connecting to the shared Team Foundation Server, the team members will then return to the simulation – this time using Visual Studio to manage the process.

- Mapping Scrum to Visual Studio 2010
- Release and Sprint work items
- Product Backlog Item (PBI) work items
- Sprint Backlog Task (SBT) work items
- Demonstration
- Simulation
- Retrospective

*This course may be run on a pre-release version of Visual Studio 2010 and/or the process template. Because of this, not all advertised capabilities may be available or demonstrated.

MODULE 4: INTRODUCTION TO THE CASE STUDY

In this module the team is introduced to their problem domain for the week. A kickoff meeting by the product owner (the instructor) will set the stage for the why and what that will take during the upcoming sprints. The team will then define the quality attributes of the project and their definition of "done." The application will be downloaded, built, and explored, so that any bugs can be discovered and reported.

- Introduction to the case study (project)
- Download the source code, build, and explore the application
- Define the quality attributes for the project
- Define "done"
- How to file effective bugs in Visual Studio 2010
- Retrospective

MODULE 5: HOTFIX

This module drops the team directly into the Brownfield experience by forcing them to analyze the existing application's architecture and code in order to locate and fix the product owner's high-priority bug. The team will learn best practices around finding, testing, fixing, validating, and closing a bug report.

- How to use Architecture Explorer to visualize and explore
- Create a unit test to validate the existence of a bug
- Find and fix the bug
- Validate and close the bug
- Retrospective

MODULE 6: SPRINT PLANNING

This short module introduces the team to release and sprint planning within Visual Studio 2010. The team will create the release work item, setting its goal, dates, and the team's capacity.

- Sprint planning meetings
- What is "potentially shippable"
- Acceptance criteria and tests
- Sprint Backlog Tasks (SBTs)
- How to create and link PBIs and SBTs
- Retrospective

At this point the team will have the knowledge of Scrum, Visual Studio 2010, and the case study application to begin developing increments of completed, potentially shippable functionality.

MODULE 7: EMERGENT ARCHITECTURE

This module introduces the architectural practices and tools a team can use to develop a valid design on which to develop new functionality. The teams will learn how Scrum supports good architecture and design practices. After the discussion, the teams will be presented with the product owner's prioritized backlog so that they may select and commit to the functionality they can deliver in this sprint.

- Architecture and Scrum
- Emergent architecture
- Design principles
- Visual Studio 2010 modeling tools
- UML and layer diagrams
- **SPRINT 1**
- Retrospective

MODULE 8: TEST DRIVEN DEVELOPMENT

This module introduces Test Driven Development as a design tool and how to implement it using Visual Studio 2010. To maximize productivity and quality, a Scrum team should setup Continuous Integration to regularly build every team member's code changes and run regression tests. Refactoring will also be defined and demonstrated in combination with Visual Studio's Test Impact Analysis to efficiently re-run just those unit tests which were impacted by refactoring.

- Continuous integration
- Test Driven Development
- Avoiding wasteful unit tests
- Refactoring
- Test Impact Analysis tool
- **SPRINT 2**
- Retrospective

MODULE 9: AGILE DATABASE TECHNIQUES

This module lets the SQL Server database developers in on a little secret – they can be agile too. By using the database projects in Visual Studio 2010, the database developers can join the development team. The team will see how you can apply agile database techniques within Visual Studio to support the SQL Server 2005/2008 development lifecycle.

- Agile database development
- Visual Studio database projects
- Importing schema and scripts
- Building and deploying
- Generating data
- Unit testing
- **SPRINT 3**
- Retrospective

MODULE 10: ACCEPTANCE TESTING

This module introduces acceptance criteria, manual tests, and the Microsoft Test and Lab Manager tool. Just because your team likes the functionality doesn't mean the product owner will. By refining acceptance criteria into manual test steps, testers can execute the tests, recording the results and report bugs in a number of ways. As the sprint completes and an increment of functionality is delivered, the team should create a branch of the codeline in order to support any hotfixes.

- Acceptance criteria
- Testing in Visual Studio 2010
- Microsoft Test and Lab Manager
- Writing and running manual tests
- Branching
- Promotion modeling
- Branching and merging
- **SPRINT 4**
- Retrospective

MODULE 11: DYSFUNCTION

Now for something completely different – the Sad path. This module introduces the many types of people, process, and tool dysfunctions that teams face in the real world. Many scenarios will be identified, along with ideas for mitigating them, to allow you and your team to move toward independence and improve your game of Scrum.

- Assessing your knowledge of Scrum
- Responsibilities of the Scrum roles
- Interpersonal dysfunctions
- Scrum-butts and flaccid Scrum
- Scrum and the Enterprise
- Course Retrospective

SCRUM FAQ

Time permitting, the instructor will present a list of frequently asked questions about Scrum, implementing Scrum, and the Scrum Developer program. The instructor will lead a discussion around these topics.

- Scrum FAQs
- Implementing Scrum in Visual Studio 2010 FAQs
- Scrum Developer FAQs

WHAT WILL BE EXPECTED OF YOU AND YOUR TEAM

This is a unique course in that it's technically-focused, team-based, and employs timeboxes. It demands that the members of the teams self-organize and self-manage their own work to collaboratively develop increments of software.

All attendees must commit to these	All teams should have these skills
• Pay attention to all lectures and demonstrations • Participate in team and group discussions • Work collaboratively with other team members • Obey the timebox for each activity • Commit to work and do your best to deliver	• Understanding of Scrum • Familiarity with Visual Studio 2010 • C#, .NET 4.0 & ASP.NET 4.0 experience • SQL Server 2008 development experience • Software testing experience

SELF-ORGANIZING TEAMS

Another unique attribute of this course is that it's a technical training class being delivered to *teams* of developers, not pairs, and not individuals. Ideally, your actual software development team will attend the training to ensure that all necessary skills are covered. However, if you wish to attend an open enrollment course alone or with just a couple of buddies, realize that you may be placed on a team with other attendees. The instructors will do their best to ensure that each team is cross-functional to tackle the case study at hand, but there are no guarantees. You may be required to try a new role, learn a new skill, or pair with somebody unfamiliar to you. This is just good Scrum.

WHO SHOULD NOT TAKE THIS COURSE

Because of the nature of this course, as explained above, certain types of people should probably not attend this course:

- Students requiring command and control style instruction – there are no prescriptive (think traditional Microsoft Learning) labs in this course
- Students who are unwilling to work collaboratively on a team
- Students who don't have any skill in any of the software development disciplines
- Students who are unwilling to work within a timebox
- Students who are unable to commit fully to their team – not only will this diminish the student's learning experience, but it will also impact their team's learning experience

FOR MORE INFORMATION

Visit www.scrum.org for more information about the Scrum Developer, the Scrum assessment, or the community of Scrum users and trainers.